Journal of Innovative Research in Computer Science & Technology (IJIRCST)
ISSN: 2347-5552, Volume-3, Issue-3, May-2015

Agent Based Approach for Discovery of Cloud Services

Rashmi Chandwadkar, M. U. Kharat

Abstract—
Cloud computing is considered as important aspect for assessing various services. There are many cloud service providers that makes provision of different services and consumers that acquire these services by mentioning requirements. Agents are required to interact with consumers and providers in order to make service negotiation decisions. Provider has to make decisions regarding which cloud service requests to accept based on availability of its resources by evaluating current demands for services as well as future demands. Agent-based approach is mainly concerned with design and development of such agents to help discover various services, negotiate with providers and consumers and composition of cloud services. The novel contributions include developing an agent-based service discovery engine, effectively adopting service negotiation methodology and composing cloud services in unified, single window solution. Service discovery agent consults ontology system for similarity reasoning and negotiation mechanism can be established between consumers and brokers for cloud service markets and between brokers and providers for cloud resource markets. The proposed system provides higher probability of getting the resources.

***Index Terms*—Cloud Computing, Cloud service discovery, service composition, service negotiation**

I. INTRODUCTION

Cloud computing provides on-demand services to the customers wherever and whenever they need. To provide services, resources such as storage, network applications, software requirements should be made available in rapid way. These computing services are transformed into a model that is delivered in same manner as that of conventional utilities. In such a model, according to the user's requirements they access services without knowledge of from where these services are delivered and hosted. Now-a-days, internet contents are assessed without knowledge of infrastructure. Infrastructure may include data centers which are monitored and maintained by content providers. Cloud computing has the capabilities of business applications that are uncovered as services and are accessed over a network. An agent takes some decision on behalf of consumer and finds out what is to be done to achieve objective. It also focuses on resource management which consists of resource pooling and resource sharing [1]. For successful communication agent should have ability to coordinate and negotiate with others. When the actions fit in well with one another we call this concept as coordination. Negotiation is when group of agents interacting and coming to a mutually acceptable solution. These agents can monitor service requests that are current and future requests as well so that they can adjust schedules and cost related to continuously changing resource demands; thereby continuously managing the resource reservation process. To manage cloud resources software agents are used. Since agents are capable of doing all above mentioned things, resource allocation and dynamically changing resource demands are adopted.

Resource pooling and sharing involve-

- Discovering cloud services by integrating various providers
- Cooperation among providers will consists of combining resources
- Contracts established between providers and consumers

In agent-based cloud computing, negotiation, cooperation and coordination protocols of agents are adopted in order to automate the activities of resource pooling and sharing in clouds. It basically aims at providing cloud services to various consumers by integration of the service providers. The service discovery will search for cloud services that match consumer's functional, technical and budgetary requirements with the provider's specifications [1]. This makes it easy for the user to discover his required services provider, verify their availability, and negotiate services cost to order services as per their needs. Agents are appropriate for decision making on behalf of both consumers who requests for services and provider who makes provision of services. Sometimes consumers need to make decision regarding selection of providers and for establishing service contracts with them, also, providers need to make decision about which service requests to accept, whether sufficient resources are available or not. Agent-based approach will help both consumers as well as providers for establishing contracts i.e. agents negotiate with providers in order to obtain "ideal" service contract match with the consumer needs [1].

Manuscript received May 23, 2015.
Rashmi Chandwadkar, Computer Department, MET BKC, Savitribai Phule Pune University, Nashik, India.
M. U. Kharat, Computer Department, MET BKC, Savitribai Phule Pune University, Nashik, India.

II. LITERATURE SURVEY

K. M. Sim presented a "Agent-based Cloud Commerce", that consist of an agent-based testbed for bolstering the Discovery of Cloud resources and SLA negotiation [11] and mentioned that in a business model for Cloud computing, for consumption of their computing capabilities users pay providers. This work proposes an agent-based testbed for strengthening the discovery of Cloud resources and SLA negotiation [8]. In the testbed, provider agents and consumer agents act as mediators between providers and consumers [1], [2], [13].

A Grid is basically collection of resources that are geographically distributed providing sharing and selection of resources dynamically depending on various aspects such as performance, QoS, capability, etc. [2], it adopts the place that many Cloud computing deployments depend on Grids, have autonomic distinctiveness that is self organizing capabilities, and some utilities. The areas that are relevant to work include: agent-based Grid resource discovery and Grid resource negotiation [2], [13].

1) Agent-based Grid resource discovery: arrangement of an algorithm for vigorously assembling agents that are capable of delivering information about distributed networked assets. Each agent occasion-ally exchanges Grid resource knowledge with other agents.
2) Grid resource negotiation: Followed a relaxed- criteria protocol for Grid resource negotiation among market-driven agents. Negotiation agents take into account the market dynamics in a Grid and are programmed to slightly relax their negotiation criteria to improve negotiation success rates.

This paper reports the ideas, design, and continuing development of an agent-based Cloud resource management testbed, and preliminary experimental results. The results consist of following aspects: The agent-based resource finding approach is generally successful in identical requests to resources according to the preferences of consumers and providers, and using the relaxed time slot negotiation protocol, consumers are generally successful in negotiating for requested time slots to use resources, and providers generally benefit from achieving superior resource utilization.

J. Kang and K. M. Sim[3] presented " Cloudle : An Agent-based Cloud Search Engine that Consults a Cloud Ontology", that consist of the search engine consulting ontology for service reasoning to find out relations of various cloud services. This paper focuses on a search engine called as Cloudle[6], [7] that is capable of offering the Cloud services based on consumer's requirements and providers specifications. This can be achieved by consulting Cloud ontology which consists of many concepts defined in it and can also be used for service reasoning [1], [13]. It is specially designed to assist users in finding Cloud services over the internet. Cloud ontology [6], [7] and three kinds of reasoning methods: 1) Similarity reasoning, 2) Compatibility reasoning and 3) Numerical reasoning are also introduced in order to provide the relevant information that approximately matches with the consumer needs. The web interface of Cloudle is also presented. The contributions of this work include by building such search engine it will make easy access for the consumers to get various cloud services in unified format [3].

This also includes that if exact matched is not found it will search relevant concept by using ontology reasoning system.

It is for the first time that agent-based service discovery system is used for retrieving information relevant to Cloud services by consulting ontology [3], [6], [7], [13]. At present, there are few Cloud service providers and there may not be many Cloud services available. However, when Cloud computing is more widely and commonly used in the near future, Cloudle can be helpful tool for Cloud users for finding Cloud services under their specific preference.

K.M. Sim and B. Shi [3] mentioned that in multi-Cloud environments, service composition should coordinate participants with their interest, service assortments should be mechanized, configure distributed services and also should be able to deal with incomplete information related to cloud providers and services provided by them. In order to compose services in various environments of multi-cloud [11], agent-based approach is used. This work consists of concurrent negotiation mechanism [9], [11] together with three classes of commitment management strategies and a utility-oriented coordination strategy for managing multiple concurrent negotiations. This work only presents the experimental results of the negotiation mechanism Grid markets [3].

Agent-based service composition [4], [10] in cloud computing is automated and supported by agents. It considers semantic aspects of web services through supporting service interaction and handling failures in order to verify and validate service compositions. However, this section is only centered on automated web service composition approaches where agents show self-organization capabilities, reaction to environment's changes, and/or make use of cost-based service selection mechanisms, given their close relation with the present work [4].

There is need for agent paradigm in Cloud computing service composition [10]. Agents are capable of solving problems independently and may collaborate with one another to achieve objectives while at the same time considering individuals goals and constraints [13]. Tools to automate Cloud resource management can be appropriately handled by the agents. This consists of resource mapping and dealing with varying requests of the consumers. Agent-based Cloud computing-the idea of adopting autonomous agents for managing Cloud resources was first introduced and proposed in [1], [13].

Cloud service composition [4], [10] may be carried out in two modalities: One-time and persistent. One-time service compositions consider Cloud resources as functions that receive consumer requirements as input parameters, and return the output accordingly. Once the output is calculated, the link between consumers and providers does not remain. On the other hand, Persistent service

compositions create a virtualized service which can be accessed by consumers for a long predefined time. This can be considered in Infrastructure as a Service. Both onetime and persistent service compositions may be amplified in horizontal and vertical dimensions [5].

III. SYSTEM ARCHITECTURE

There are various Cloud Service providers. Each service provider has their own service packages which may or may not be known to user. The problem to the user is to discover his required services provider, verify their availability, and negotiate services cost to order services. This tends to be time consuming and ineffective while erroneous in choosing services. Agents can help in better way to deal with such problem by taking decisions relevant to negotiation of services. Proposed system consists of development of service discovery engine, establishment of service level agreements between various consumers and providers via agents and providing a single window solution to the consumers based on their requirements. Also, negotiation issues of QoS and time slot matching is addressed there by handling failure of the system. Following Fig.1 is the system architecture:

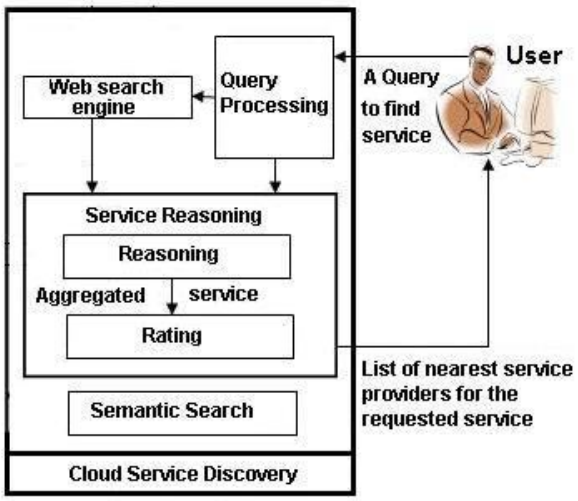

Fig 1: System Architecture

Consumers enter their queries for cloud services using web interface. The inputs to web interface consist of a consumer's functional, technical, and budgetary requirements for a cloud service. The SDA has four sub modules:

- a query processor
- a service reasoning module
- a price and time slot matching(rating) module a service rating module

Using the query processor, the SDA extracts essential keywords such as cloud concepts and the price and schedule specifications from consumers' queries. The SDA consults cloud ontology, to reason about the similarity between a consumer's functional and technical requirements and providers' functional and technical specifications of services [1]. Cloud ontology consists of a set of cloud computing concepts and the interrelationships among cloud concepts to facilitate the reasoning about cloud services. Using the cloud ontology, the SDA carries out: 1) similarity reasoning, 2) compatibility reasoning, and 3) numerical reasoning. In addition, the SDA states matching level of the price constraints of consumers and providers, using the price and time slot module. The service rating module consists of rating given to each provider acceding to the similarity between specifications of consumer and provider [1]. The resulting data has list of cloud service providers that are ordered in terms of matching similarities of various aspects specified by consumer and service providers.

IV. IMPLEMENTATION DETAILS

This section gives idea about the strategy of implementation. Basically, the system consists of accepting various requirements from consumer and processing it in order to obtain the cloud services accordingly. There are important modules of system that needs to be considered. These modules are: register provider, Search engine, and inline workflow and rating engine. Each module has its significance.

A. Implementation Modules

1) Register Provider: Provider will register to the search engine database catalog. Basically, this will consist of extracting the API's of the cloud service provider and storing them into repository, thereby maintaining data of providers.

2) Search Engine: Discover cloud services against various requirements of the consumer. These requirements are mentioned in the form of technical, functional and budgetary requirements. Search engine has to discover the services that match consumer requirements and provider's specifications along with price and time slot matching parameter. It uses algorithm for searching such web services. It also considers Quality of Service information for better outcome.

B. Algorithm

Algorithm: Cloud services matching algorithm

Following is the specified version of service selection algorithm where the leftmost numbers denote the line numbers. When the discovery request is received, it executes fMatch (line 3) which returns a list of providers PL1 that meet the consumers' requirements. If requirements are not satisfied then its similarity of consumer requirement terms are calculated and provider list is returned. If services are available with provider these are delivered to consumers (line 6). In case the services are unavailable then next provider is searched in the PL (line 6). If only one service satisfies the selection criteria, it returns this service to the customer.

/* Cloud services selection algorithm */

Parameters:
PL: list of providers
fMatch: function that matches the consumer requirements and provider specifications
cReq: consumer requirements
pSpec: provider specification

psimList: list of similarity matching of services
Input: Set of consumer requirements
Output: List of Cloud Services

1)//Get requirements from consumer
cReq = getRequirements(functionalReq, TechnicalReq, BudgetaryReq)

2) // number of providers that offers cloud services pSpec = getProviderList(functionalSpec, technicalSpec, budgetarySpec)

3) //find cloud services that match various requirements and specifications
for each provider in PL
  if(fMatch (cReq, pSpec))
    PL.add(provider)
  return PL goto step 5

4) // calculate similarity of consumer requirements terms
if(PL.empty())
  for each provider
    if(fsimMatch(cReq,pSpec))
      psimList.add(provider)
  return psimList

5) Select the appropriate provider

6) if(services available with provider)
    Provide and deliver services
  goto step 8
  else
    searchNextProvider()f for each provider in PL
  if (services available)
    goto step 5
  else
    searchNextProvider()

7) if(services not available) then
    suggestion of PL and psimList or modify query
    goto step 3
  else
    fail

8) stop

C. Mathematical Model

A system should aim to develop an agent-based search engine for supporting cloud service discovery and negotiating with providers in order to acquire services.

- Consumers specify requirements
- Service Discovery search for services
- Price and time Slot matching
- Give cloud services list

Let S be an Agent-based System that accepts requirements of Consumer and matches with provider's specifications. Where,
S: is a system which holds list of
A: set of Agents
U: set of services to be provided
C: set of Consumers
P: set of providers
N: Natural numbers

$S = \{C, P, U, A\}$
Where,
$C = \{c_i \mid c_i \in S \wedge i \in N\}$
$P = \{p_i \mid p_i \in A \wedge i \in N\}$
$U = \{u_i \mid u_i \in P \wedge i \in N\}$
$A = \{a_i \mid a_i \in S \wedge i \in N\}$
Relation(R): Let R be a set of relationship between consumers, agents and providers.
$R = \{r_i \mid i \leq N\}$
$r_1 = \{(A, C_i) \mid A \in S \wedge C_i \in S\}$
$r_2 = \{(A, P_i) \mid A \in S \wedge P_i \in A\}$
$r_3 = \{(P_i, C_i) \mid P_i \in A \wedge C_i \in S\}$
$r_4 = \{(P_i, U_i) \mid P_i \in A \wedge U_i \in P\}$
$r_5 = \{(C_i, U_i) \mid C_i \in S \wedge U_i \in A\}$
$r_6 = \{(A, U_i) \mid A \in S \wedge U_i \in P\}$

Consider the following relationship between various functions and its dependencies
Fn1: Set of consumers requesting cloud services
Fn2: Agent searching for various cloud services
Fn3: Set of Providers that have registered themselves to agents.

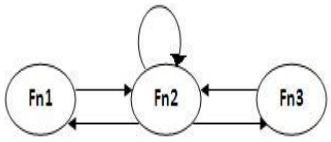

Fig 2: State chart diagram

D. Outcome Analysis

In experimentation, data set of various providers is taken into consideration. The current stage performs the refinement of providers based on selection of services. The input to the system consists of various parameters such as RAM size, CPU speed, hard disk drive size, etc. To search providers, these are registered to the search engine database catalog. The expected performance measure is based on success rate of the cloud service discovery agent that consults cloud ontology. The outcome provides cloud services in a single window solution i.e., the services are consolidated so that it is easy for consumer to assess these services. Each entry is a specification of service from cloud service providers. Taking cloud computing service as an example, Table 1 shows a requirement of consumer such as 5 nCPUs, 2G RAM and some CPU speed.

Table 1: Partial view of consumer requirements

RAM	vCPU	CPU Speed	OS Platform	OS Type	Disk	Price
2Gb	5	1.7 ghz	Windows	Windows	80	$0.93/hr
2Gb	3	2.1 ghz	Linux	Fedora	240	$1.44/hr
4Gb	2	1.7 ghz	Windows	Windows	40	$0.54/hr
8Gb	5	2.3 ghz	Windows	Windows	10	$0.03/hr
2Gb	2	2.3 ghz	Linux	Red Hat	20	$0.06/hr
1Gb	5	1.8 ghz	Linux	Fedora	10	$0.03/hr
2Gb	1	1.6 ghz	Windows	Windows	40	$0.54/hr
4Gb	2	1.7ghz	Linux	Red Hat	20	$0.06/hr
16Gb	24	2.5ghz	Windows	Windows	320	$3.56/hr

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications [12]. Each instance type includes one or more instance sizes, allowing you to scale your resources to the requirements of your target workload. Following table contains sample of Amazon instance type [12]:

Table 2: Instance types matrix of Amazon (ref. from [12])

Instance Type	vCPU	Memory (GB)	Storage (GB)	Clock Speed	Physical Processor
t2.micro	1	1	EBS only	2.5	Intel Xeon family
t2.small	1	2	EBS only	2.5	Intel Xeon family
t2.medium	2	4	EBS only	2.5	Intel Xeon family
m3.medium	1	3.75	1 X 4 SSD	2.5	Intel Xeon E5-2670 v2*
m3.large	2	7.5	1 X 32 SSD	2.5	Intel Xeon E5-2670 v2*
m3.2.xlarge	8	30	2 X 80 SSD	2.5	Intel Xeon E5-2670 v3

T2 instances are a good choice for workloads that don't use the full CPU often or consistently [12]. High Frequency Intel Xeon Processors operating at 2.5GHz with Turbo up to 3.3GHz. Lowest-cost general purpose instance type, and Free Tier eligible (t2.micro only)[12].The next family includes the M3 instance types and provides a balance of compute, memory, and network resources, and it is a good choice for many applications.

The following table shows statistic view of outcome of the system. It consists of matching percentage of data content based on similarity between consumer and provider specifications.

Table 3: Result set for similarity matching

RAM	vCPU	CPU Speed	OS Platform	OS Type	Disk	Price	Similar Match (%)
2Gb	5	1.7 ghz	Windows	Windows	80	$0.93/hr	60
2Gb	3	2.1 ghz	Linux	Fedora	240	$1.44/hr	33.6
4Gb	2	1.7 ghz	Windows	Windows	40	$0.54/hr	45.5
8Gb	5	2.3 ghz	Windows	Windows	10	$0.03/hr	56
2Gb	2	2.3 ghz	Linux	Red Hat	20	$0.06/hr	52
1Gb	5	1.8 ghz	Linux	Fedora	10	$0.03/hr	62
2Gb	1	1.6 ghz	Windows	Windows	40	$0.54/hr	20
4Gb	2	1.7ghz	Linux	Red Hat	20	$0.06/hr	10
16Gb	24	2.5ghz	Windows	Windows	320	$3.56/hr	20
16Gb	8	2.5ghz	Windows	Windows	240	$2.56/hr	10

.

V. CONCLUSION

Agent based cloud computing system enhances the overall functioning of the cloud system. The consumers and the providers find it difficult to match the requirements and specification of one another. The consumer has to discover required services, verify their availability and negotiate services cost with the providers. This tends to be time consuming and effective. Agents are capable of providing services in unified form. Several novel approaches are used for cloud resource management such as service discovery, service negotiation and service composition. Effectiveness of the modified implementation consists of providing higher probability of getting the services as needed. Using the Cloud Service Discovery system, expected results indicate that consumer finds easy to discover services in a single window solution.

ACKNOWLEDGMENT

The authors wish to thank MET's Institute of Engineering Bhujbal Knowledge City, Nashik, India for providing lab facilities. Authors are also thankful to K. M. Sim, "Agent-Based Cloud Computing" for their work in this area. The mentioned paper is major guideline for this work.

REFERENCES

[1] K. M. Sim," Agent-based Cloud Computing" Special Issue on Cloud Computing in IEEE Transactions on Services Computing. (IEEE Computer Society) . Published online for early access. DOI: 10.1109/TSC.2011.52

[2] K.M. Sim,"Agent-Based Cloud Commerce",Proc. IEEE Intl Conf. Indus-trial Eng. and Eng. Management, pp. 717-721, 2009.

[3] J. Kang and K.M. Sim, "Cloudle: An Agent-Based Cloud Search Engine that Consults a Cloud Ontology," Proc. Intl Conf. Cloud Computing and Virtualization, pp. 312-318, May 2010.

[4] K.M. Sim and B. Shi, "Concurrent Negotiation and Coordination for Controlling Grid Resource Co-Allocation", IEEE Trans. Systems, Man and Cybernetics, Part B, vol. 40, no. 2, pp. 753-766, June 2010.

[5] J.O. Gutierrez-Garcia and K.M. Sim, "Agent-Based Cloud Service Com-position",Proc. 2010 Conf. Grid and Distributed Computing, Dec. 2010.

[6] T. Andreasen, H. Bulskov, and R. Kanppe, "From Ontology over Similarity to Query Evaluation", Proc. Second Intl Conf. Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems (ODBASE), Nov. 2003.

[7] L. Youseff, M. Butrico, and D. Da Silva, "Toward a Unified Ontology of Cloud Computing",Proc. Grid Computing Environments Workshop (GCE 08), pp. 1-10, 2008.

[8] K.M. Sim and C.Y. Choi, "Agents that React to Changing Market Situations", IEEE Trans. System, Man, Cybernetics B, vol. 33, no. 2, pp. 188-201, Apr. 2003.

[9] K.M. Sim and B. Shi, "Concurrent Negotiation and Coordination for Controlling Grid Resource Co-Allocation", IEEE Trans. Systems, Man and Cybernetics, Part B, vol. 40, no. 2, pp. 753-766, June 2010.
[10] J.O. Gutierrez-Garcia and K.M. Sim, "Self-Organizing Agents for Service Composition in Cloud Computing",Proc. Second IEEE Intl Conf. Cloud Computing Technology and Science, 2010.
[11] T.D. Nguyen and N.R. Jennings, "Managing Commitments in Multiple Concurrent Negotiations", Intl J. Electronic Commerce Research and Applications, vol. 4, no. 4, pp. 362-376, 2005.
[12] Amazon: http:/www.aws.amazon.com/ec2/instance-types/
[13] Rashmi Chandwadkar and M. U. Kharat, "Review on Agent Based Cloud Computing," Proc. International Journal of Computer Applications, Conf. Innovations and Trends in Computer and Communication Engineering, pp. 0975-8887, Dec 2014.